On the use of polynomial matrix approximant in the block Wiedemann algorithm

Pascal Giorgi

Laboratoire LIP,
ENS Lyon, France

Symbolic Computation Group,
School of Computer Science,
University of Waterloo, Canada

in collaboration with C-P. Jeannerod and G. Villard ENS-Lyon (France)

June 6, CMS/CSHPM Summer 2005 Meeting
Mathematics of Computer Algebra and Analysis

Motivations

Large sparse linear systems are involved
in many mathematical applications

over a field :

- integers factorization [Odlyzko 1999]],
- discrete logarithm [Odlyzko 1999 ; Thomé 2003],

over the integers :

- number theory [Cohen 1993],
- group theory [Newman 1972],
- integer programming [Aardal, Hurkens, Lenstra 1999]

Solving sparse system over finite fields

Matrices arising in pratice have dimensions around few millions with few millions of non zero entries

The use of classic Gaussian elimination is proscribed due to fill-in

Solving sparse system over finite fields

Matrices arising in pratice have dimensions around few millions with few millions of non zero entries

The use of classic Gaussian elimination is proscribed due to fill-in
$\Longrightarrow$ algorithms must preserve the sparsity of the matrix

Solving sparse system over finite fields

Matrices arising in pratice have dimensions around few millions with few millions of non zero entries

The use of classic Gaussian elimination is proscribed due to fill-in
$\Longrightarrow$ algorithms must preserve the sparsity of the matrix

Iteratives methods revealed successful over a finite field :

- Krylov/Wiedemann method [Wiedemann 1986]
- conjugate gradient [Lamacchia, Odlyzko 1990],
- Lanczos method [Lamacchia, Odlyzko 1990 ; Lambert 1996],
- block Lanczos method [Coppersmith 1993, Montgomery 1995]
- block Krylov/Wiedemann method [Coppersmith 1994, Thomé 2002]

Scheme of Krylov/Wiedemann method

Let $A \in \mathbb{F}^{\mathrm{N}\times\mathrm{N}}$ of full rank and $b \in \mathbb{F}^{\mathrm{N}}$. Then $x = A^{-1}b$ can be expressed as a linear combination of the Krylov subspace $\{b, Ab, ..., A^N b\}$

Scheme of Krylov/Wiedemann method

Let $A \in \mathbb{F}^{\mathrm{N}\times\mathrm{N}}$ of full rank and $b \in \mathbb{F}^{\mathrm{N}}$. Then $x = A^{-1}b$ can be expressed as a linear combination of the Krylov subspace $\{b, Ab, ..., A^N b\}$

Let $\Pi^{Ab}(\lambda) = c_0 + c_1\lambda + ... + \lambda^d \in \mathbb{F}[\lambda]$ be the minimal polynomial of the sequence $\{A^i b\}_{i=0}^{\infty}$

Scheme of Krylov/Wiedemann method

Let $A \in \mathbb{F}^{N \times N}$ of full rank and $b \in \mathbb{F}^{N}$. Then $x = A^{-1}b$ can be expressed as a linear combination of the Krylov subspace $\{b, Ab, ..., A^N b\}$

Let $\Pi^{Ab}(\lambda) = c_0 + c_1\lambda + ... + \lambda^d \in \mathbb{F}[\lambda]$ be the minimal polynomial of the sequence $\{A^i b\}_{i=0}^{\infty}$

$$A \quad \times \quad \frac{-1}{c_0}(c_1 b + c_2 Ab + ... + A^{d-1} b) \quad = \quad b$$

Scheme of Krylov/Wiedemann method

Let $A \in \mathbb{F}^{\mathrm{N}\times\mathrm{N}}$ of full rank and $b \in \mathbb{F}^{\mathrm{N}}$. Then $x = A^{-1}b$ can be expressed as a linear combination of the Krylov subspace $\{b, Ab, ..., A^N b\}$

Let $\Pi^{Ab}(\lambda) = c_0 + c_1\lambda + ... + \lambda^d \in \mathbb{F}[\lambda]$ be the minimal polynomial of the sequence $\{A^i b\}_{i=0}^{\infty}$

$$A \quad \times \quad \underbrace{\frac{-1}{c_0}(c_1 b + c_2 Ab + ... + A^{d-1}b)}_{x} \quad = \quad b$$

Scheme of Krylov/Wiedemann method

Let $A \in \mathbb{F}^{\mathrm{N}\times\mathrm{N}}$ of full rank and $b \in \mathbb{F}^{\mathrm{N}}$. Then $x = A^{-1}b$ can be expressed as a linear combination of the Krylov subspace $\{b, Ab, ..., A^N b\}$

Let $\Pi^{Ab}(\lambda) = c_0 + c_1\lambda + ... + \lambda^d \in \mathbb{F}[\lambda]$ be the minimal polynomial of the sequence $\{A^i b\}_{i=0}^{\infty}$

$$A \quad \times \quad \underbrace{\frac{-1}{c_0}(c_1 b + c_2 Ab + ... + A^{d-1} b)}_{x} \quad = \quad b$$

Choose $u \in \mathbb{F}^{\mathrm{N}}$ uniformaly and randomly and compute the minimal polynomial $\Pi^{u,Ab}$ of the scalar sequence $\{u^T A^i b\}_{i=0}^{\infty}$.

with probability greater than $1 - \frac{\deg(\Pi^{Ab})}{Card(\mathbb{F})}$ we have $\Pi^{Ab} = \Pi^{u,AB}$.

Wiedemann algorithm

Three steps :

1. compute $2N + \epsilon$ elements of the sequence $\{u^T A^i b\}_{i=0}^{\infty}$.
2. compute the minimal polynomial of the scalar sequence.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1} b\}$.

Wiedemann algorithm

Three steps :

1. compute $2N + \epsilon$ elements of the sequence $\{u^T A^i b\}_{i=0}^{\infty}$.
2. compute the minimal polynomial of the scalar sequence.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

Wiedemann algorithm

Three steps :

1. compute $2N + \epsilon$ elements of the sequence $\{u^T A^i b\}_{i=0}^{\infty}$.
2. compute the minimal polynomial of the scalar sequence.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

step 1 : $2N$ matrix-vector products + $2N$ dot products

$\Longrightarrow$ cost $2N\gamma + O(N^2)$ field operations

Wiedemann algorithm

Three steps :

1. compute $2N + \epsilon$ elements of the sequence $\{u^T A^i b\}_{i=0}^{\infty}$.
2. compute the minimal polynomial of the scalar sequence.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

step 1 : $2N$ matrix-vector products + $2N$ dot products
$\Longrightarrow$ cost $2N\gamma + O(N^2)$ field operations

step 2 : use of Berlekamp/Massey algorithm [Berlekamp 1968, Massey 1969]
$\Longrightarrow$ cost $O(N^2)$ field operations

Wiedemann algorithm

Three steps :

1. compute $2N + \epsilon$ elements of the sequence $\{u^T A^i b\}_{i=0}^{\infty}$.
2. compute the minimal polynomial of the scalar sequence.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

step 1 : $2N$ matrix-vector products + $2N$ dot products
$\Longrightarrow$ cost $2N\gamma + O(N^2)$ field operations

step 2 : use of Berlekamp/Massey algorithm [Berlekamp 1968, Massey 1969]
$\Longrightarrow$ cost $O(N^2)$ field operations

step 3 : $d - 1$ matrix-vector products + $d - 1$ vectors operations
$\Longrightarrow$ cost at most $N\gamma + O(N^2)$ field operations

total cost of $O(N\gamma + N^2)$ fied operations with $O(N)$ additional space

Block Wiedemann method

Replace the projection vectors by blocks of vectors.

Let $U \in \mathbb{F}^{\mathrm{m}\times\mathrm{N}}$ and $V = [b \quad \bar{V}] \in \mathbb{F}^{\mathrm{N}\times\mathrm{n}}$.
We now consider the matrix sequence $\{UA^iV\}_{i=0}^{\infty}$.

Main interest :

- parallel coarse and fine grain implementation (on columns of V),
- better probability of success [Villard 1997],
- $(1+\epsilon)N$ matrix-vector products (sequential) [Kaltofen 1995].

Difficulty :

minimal generating matrix polynomial of a matrix sequence.

Minimal generating matrix polynomial

Let $\{S_i\}_{i=0}^{\infty}$ be a $m \times m$ matrix sequence.

Let $P \in \mathbb{F}^{\mathrm{m}\times\mathrm{m}}[\lambda]$ be minimal with degree k s.t.

$$\forall j > 0 : \quad \sum_{i=0}^{k} S_{i+j} P_{[i]} = 0^{m\times m}$$

the cost to compute P is :

- $O(m^3 k^2)$ field operations [Coppersmith 1994],
- $O\tilde{}(m^3 k \log k)$ field operations [Beckermann, Labahn 1994 ; Thomé 2002],
- $O\tilde{}(m^\omega k \log k)$ field operations [Giorgi, Jeannerod, Villard 2003].

where ω is the exponent in the complexity of matrix multiplication

Minimal generating matrix polynomial

Let $\{S_i\}_{i=0}^{\infty}$ be a $m \times m$ matrix sequence.

Let $P \in \mathbb{F}^{m \times m}[\lambda]$ be minimal with degree k s.t.

$$\forall j > 0 : \quad \sum_{i=0}^{k} S_{i+j} P_{[i]} = 0^{m \times m}$$

the cost to compute P is :

- $O(m^3 k^2)$ field operations [Coppersmith 1994],
- $\tilde{O}(m^3 k \log k)$ field operations [Beckermann, Labahn 1994 ; Thomé 2002],
- $\tilde{O}(m^\omega k \log k)$ field operations [Giorgi, Jeannerod, Villard 2003].

where ω is the exponent in the complexity of matrix multiplication

Latter complexity is based on σ-*bases* computation with :

- divide and conquer approach (idea from [Beckermann, Labahn 1994])
- matrix product-based Gaussian elimination [Ibarra et al 1982]

Minimal approximant basis : σ-*basis*

Problem :

Given a matrix power series $G \in \mathbb{F}^{m \times n}[[\lambda]]$ and an approximation order d ; find the minimal nonsingular polynomial matrix $M \in \mathbb{F}^{m \times m}[\lambda]$ s.t.

$$MG = \lambda^d R \in \mathbb{F}^{m \times n}[[\lambda]]$$

M is called an **order d σ-bases of G**.

Minimal approximant basis : σ-*basis*

Problem :
Given a matrix power series $G \in \mathbb{F}^{m\times n}[[\lambda]]$ and an approximation order d ; find the minimal nonsingular polynomial matrix $M \in \mathbb{F}^{m\times m}[\lambda]$ s.t.

$$MG = \lambda^d R \in \mathbb{F}^{m\times n}[[\lambda]]$$

M is called an **order d σ-bases of G**.

minimality :

Let $f(\lambda) = G(\lambda^n)[1, \lambda, \lambda^2, ..., \lambda^n]^T \in \mathbb{F}[[\lambda]]^m$

every $v \in \mathbb{F}^{1\times m}[\lambda]$ such that

$$v(\lambda^n)f(\lambda) = \lambda^r w(\lambda) \in \mathbb{F}[[\lambda]] \text{ with } r \geq nd$$

has a unique decomposition

$$v = \sum_{i=1}^{m} c^{(i)} M^{(i,*)} \text{ with } \deg c^{(i)} + \deg M^{(i,*)} \leq \deg v$$

Sketch of the reduction

divide and conquer :

[Beckermann, Labahn 1994 : theorem 6.1]
Given M' and M'' two order **d/2** σ-**basis** of respectively G and $\lambda^{\frac{-d}{2}} M' G$.
The polynomial matrix $M = M'M''$ is an order **d** σ-**bases** of G.

base case (order 1 σ-basis) :

- compute $\Delta = G \mod \lambda$,
- compute the LSP-factorization of $\pi\Delta$, with π a permutation,
- return $M = DL^{-1}\pi$ where D is a diagonal matrix (with 1's and λ's)

Sketch of the reduction

divide and conquer :

[Beckermann, Labahn 1994 : theorem 6.1]

Given M' and M'' two order **d/2** σ-**basis** of respectively G and $\lambda^{\frac{-d}{2}} M' G$.
The polynomial matrix $M = M' M''$ is an order **d** σ-**bases** of G.

base case (order 1 σ-basis) :

- compute $\Delta = G \mod \lambda$,
- compute the LSP-factorization of $\pi\Delta$, with π a permutation,
- return $M = DL^{-1}\pi$ where D is a diagonal matrix (with 1's and λ's)

cost :

- $C(m, n, d) = 2C(m, n, d/2) + 2MM(m, d/2)$
- $C(m, n, 1) = O(MM(m))$

$\Longrightarrow$ reduction to polynomial matrix multiplication

σ-*bases* and minimal generating matrix polynomial

Considering the matrix power series $G(\lambda) = \sum_{i=0}^{\infty} UA^i V \lambda^i \in \mathbb{F}^{\mathrm{m}\times\mathrm{n}}[[\lambda]]$

Let $P, T \in \mathbb{F}^{\mathrm{n}\times\mathrm{n}}$ and $Q, S \in \mathbb{F}^{\mathrm{m}\times\mathrm{m}}[\lambda]$ defining the right $2N/m$ σ-bases

$$\begin{bmatrix} G & -I_m \end{bmatrix} \begin{bmatrix} P & S \\ Q & T \end{bmatrix} = \lambda^{2N/m} R \in \mathbb{F}^{\mathrm{m}\times(\mathrm{m}+\mathrm{n})}[[\lambda]]$$

such that $\deg P > \deg Q$ and P is full rank.

σ-*bases* and minimal generating matrix polynomial

Considering the matrix power series $G(\lambda) = \sum_{i=0}^{\infty} UA^iV\lambda^i \in \mathbb{F}^{m\times n}[[\lambda]]$

Let $P, T \in \mathbb{F}^{n\times n}$ and $Q, S \in \mathbb{F}^{m\times m}[\lambda]$ defining the right $2N/m$ σ-bases

$$\begin{bmatrix} G & -I_m \end{bmatrix} \begin{bmatrix} P & S \\ Q & T \end{bmatrix} = \lambda^{2N/m} R \in \mathbb{F}^{m\times(m+n)}[[\lambda]]$$

such that $\deg P > \deg Q$ and P is full rank.

The reversal matrix polynomial of P according to its column degrees define a right minimal generating matrix polynomial for the matrix sequence $\{UA^iV\}_{i=0}^{\infty}$

Proof :

$$\forall k \in \{\deg P, ..., 2N/m\} \quad \sum_{i=0}^{\deg P} G^{(k-i)} P^{(i)} = 0^{m\times n}$$

Block Wiedemann algorithm

Three steps :

1. compute $2N/m + \epsilon$ elements of the sequence $\{U^T A^i V\}_{i=0}^{\infty}$.
2. compute the right minimal matrix polynomial through σ-bases.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Block Wiedemann algorithm

Three steps :

1. compute $2N/m + \epsilon$ elements of the sequence $\{U^T A^i V\}_{i=0}^{\infty}$.
2. compute the right minimal matrix polynomial through σ-bases.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

Block Wiedemann algorithm

Three steps :

1. compute $2N/m + \epsilon$ elements of the sequence $\{U^T A^i V\}_{i=0}^{\infty}$.
2. compute the right minimal matrix polynomial through σ-bases.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

step 1 : *with m processors*

$\Longrightarrow$ cost $2N\gamma/m + O(N^2)$ field operations

Block Wiedemann algorithm

Three steps :

1. compute $2N/m + \epsilon$ elements of the sequence $\{U^T A^i V\}_{i=0}^{\infty}$.
2. compute the right minimal matrix polynomial through σ-bases.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

step 1 : *with m processors*

$\Longrightarrow$ cost $2N\gamma/m + O(N^2)$ field operations

step 2 : *with 1 processors*

$\Longrightarrow$ cost $O\tilde{}(m^{\omega-1}N)$ field operations

Block Wiedemann algorithm

Three steps :

1. compute $2N/m + \epsilon$ elements of the sequence $\{U^T A^i V\}_{i=0}^{\infty}$.
2. compute the right minimal matrix polynomial through σ-bases.
3. compute the linear combination of $\{b, Ab, ..., A^{d-1}b\}$.

Let γ be the cost of applying a vector to A.

step 1 : *with m processors*
$\implies$ cost $2N\gamma/m + O(N^2)$ field operations

step 2 : *with 1 processors*
$\implies$ cost $\tilde{O}(m^{\omega-1}N)$ field operations

step 3 : *with m processors*
$\implies$ cost at most $N\gamma/m + O(N^2)$ field operations

total of $O(N\gamma/m) + O(N^2) + \tilde{O}(n^{\omega-1}N)$ fied op. *with m processors*

Implementation within LinBox library

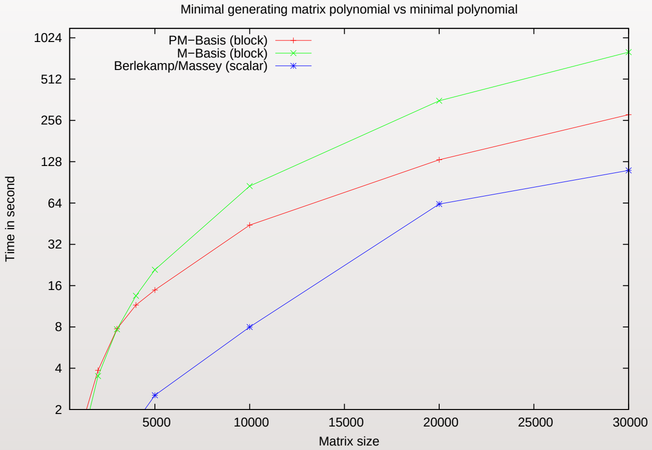
- LinBox project (Canada-France-USA) : `www.linal.org`

- Generic implementation with respect to : finite field, blackbox.

- σ-basis implementation :
  - hybrid dense linear algebra over finite field [Dumas, Giorgi, Pernet 2004]
  - polynomial matrix multiplication :
    Karatsuba algorithm + BLAS-based matrix multiplication
  - Karatsuba polynomial matrix middle product [Hanrot et al. 2003]

Performances : minimal generating matrix polynomial

- over GF(17), matrix sparsity is 99%, block dimension is 20



$\underline{N = 30\,000}$: practical block/scalar $\approx O(1)$

scalar sequence computation : $\approx 12.4h$

Conclusions

- $O\tilde{}(n^\omega d)$ algorithm for Pade approximation problem.

$\rightarrow$ advantage for solving sparse linear system (block Wiedemann)

$\rightarrow$ $O\tilde{}(n^\omega d)$ algorithm for column reduction [Giorgi, Jeannerod, Villard 2003]

- still unclear : characteristic polynomial, Hermite and Frobenius form.

Into practice :

- fast polynomial matrix multiplication
[Cantor, Kaltofen 1991 ; Bostan, Schost 2004]

- compare LinBox with Magma (*implementation of Thomé algorithm*)

- Specialization for `GF(2)` and parallel implementation (SMP, grid)